Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claim 1 (currently amended): A method for managing type information for operands, the method comprising:

accomplishing the following results through execution of a single register instruction in a register of a processor:

adding an operand tag to a tag stack; and

updating a stack pointer for the tag stack by movement of the stack pointer to recognize the addition of the operand tag to the tag stack, the stack pointer stored in the tag stack and implicitly encoded, wherein a bit position of the stack pointer is to indicate a depth of the tag stack.

Claim 2 (previously presented): A method according to claim 1, wherein the single register instruction comprises a shift instruction.

Claim 3 (original): A method according to claim 2, wherein the shift instruction comprises a rotate instruction.

Claim 4 (previously presented): A method according to claim 1, further comprising:

accomplishing the following results through execution of one register instruction:

removing an operand tag from the tag stack; and

updating the stack pointer for the tag stack to recognize the removal of the operand tag from the tag stack.

Claim 5 (previously presented): A method according to claim 4, wherein the one register instruction comprises a shift right instruction.

Claim 6 (currently amended): A method for managing type information for operands, the method comprising:

shifting a bit value of 1 into a first significant bit of a register and shifting all bits of the register in a first direction, in conjunction with creation of a reference operand; and

shifting a bit value of 0 into the first significant bit of the register and shifting all bits of the register in the first direction, in conjunction with creation of a non-reference operand.

Claim 7 (currently amended): A method according to claim 6, wherein:

the register serves as a tag stack register, the tag stack register to be used for storing a stack of operand tags, each operand tag to indicate whether a corresponding operand on an operand stack is to be treated as a reference operand or a non-reference operand; and

the method further comprises initializing the tag stack register by:

assigning a low order bit of the tag stack register to a value of 0 to implicitly encode a stack pointer, wherein a bit position of the stack pointer is to indicate a depth of the stack of operand tags; and

assigning other bits of the tag stack register to a value of 1.

Claim 8 (currently amended): A method according to claim 6, further comprising:

using a shift left ~~operations~~ operation to shift a bit ~~values~~ value into a low order bit of the register and shift a bit value of preceding order bits of the register into succeeding order bits of the register in response to an ~~operands~~ operand being added to an operand stack.

Claim 9 (previously presented): A method according to claim 6, further comprising:

right shifting bit values in the register in conjunction with removal of an operand, the operand being one of the reference operand and the non-reference operand.

Claim 10 (original): A method according to claim 9, further comprising:

shifting the bit value of 1 into a high order bit of the register in conjunction with removal of the operand.

Claim 11 (currently amended): A method according to claim 7 ~~6, wherein:~~

~~the register serves as a tag stack register, the tag stack register to be used for storing a stack of operand tags, each operand tag to indicate whether a corresponding operand on an operand stack is to be treated as a reference operand or a non-reference operand; and~~

~~the method~~ further ~~comprises~~ comprising:

treating a highest order bit with the value of 0 in the tag stack register as [[a]] the stack pointer; and

determining [[a]] the depth of the stack of operand tags, based at least in part on a location of the stack pointer.

Claim 12 (currently amended): A processing system with control logic for managing type information for operands, the processing system comprising:

a processor;

a machine-accessible storage medium responsive to the processor; and

instructions in the machine-accessible storage medium, the instructions to implement at least part of a virtual machine when executed by a processing system, the virtual machine to accomplishing the following results through execution of a single register instruction:

adding an operand tag to a first order position of a tag stack to indicate a type of operand added to an operand stack, wherein the operand tag is of a first value to indicate a reference type and of a second value to indicate a non-reference type; [[and]]

shifting a previous value stored in the first order position and all other order positions of the tag stack in a first direction; and

updating a stack pointer for the tag stack by movement of the stack pointer to recognize the addition of the operand tag to the tag stack, the stack pointer stored in the tag stack and implicitly encoded, wherein a bit order position of the stack pointer is to indicate a depth of the tag stack.

Claim 13 (previously presented): A processing system according to claim 12, wherein the single register instruction to be used by the virtual machine to add the operand tag to the tag stack and to update the stack pointer comprises a shift instruction.

Claim 14 (original): A processing system according to claim 13, wherein the shift instruction comprises a rotate instruction.

Claim 15 (previously presented): A processing system according to claim 12, the virtual machine further to accomplish the following results through execution of one register instruction:

removing an operand tag from the tag stack; and

updating the stack pointer for the tag stack to recognize the removal of the operand tag from the tag stack.

Claim 16 (original): A processing system according to claim 12 wherein the processor supports a little-endian byte order.

Claim 17 (currently amended): An apparatus containing control logic for managing type information for operands, the apparatus comprising:

a machine-accessible storage medium; and

instructions in the machine-accessible storage medium, the instructions to implement at least part of a virtual machine when executed by a processing system, the virtual machine to accomplishing the following results through execution of a single register instruction:

adding an operand tag to a first order position of a tag stack to indicate a type of operand added to an operand stack, wherein the operand tag is of a first value to indicate a reference type and of a second value to indicate a non-reference type; [[and]]

shifting a previous value stored in the first order position and all other order positions of the tag stack in a first direction; and

updating a stack pointer for the tag stack by movement of the stack pointer to recognize the addition of the operand tag to the tag stack, the stack pointer stored in the tag stack and implicitly encoded, wherein a bit order position of the stack pointer is to indicate a depth of the tag stack.

Claim 18 (previously presented): An apparatus according to claim 17, wherein the single register instruction to be used by the virtual machine to add the operand tag to the tag stack and to update the stack pointer comprises a shift instruction.

Claim 19 (original): An apparatus according to claim 18, wherein the shift instruction comprises a rotate instruction.

Claim 20 (previously presented): An apparatus according to claim 17, the virtual machine further to accomplish the following results through execution of one register instruction:

removing an operand tag from the tag stack; and

updating the stack pointer for the tag stack to recognize the removal of the operand tag from the tag stack.